

LabMaker.Tech

***MakerIoT – Da ideia ao protótipo***

## Bruno Ferreira

- Físico pela UNICAMP
- Educador Maker
- Membro do Laboratório Hacker de Campinas – LHC
- Professor Robótica e Física no SESI SP
- CPO da LabMaker.Tech



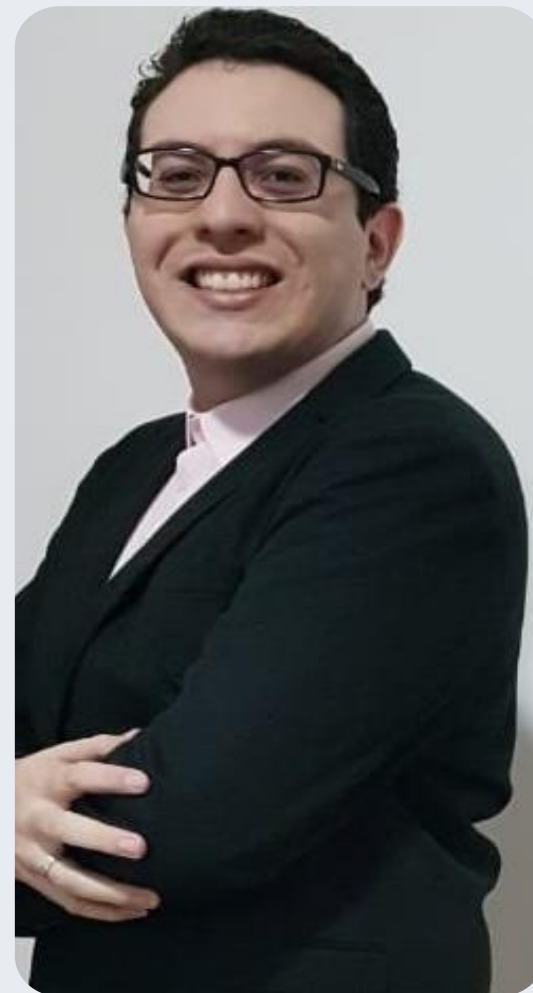
/agrofoglio

# Vinícius Souza

- Técnico em mecatrônica
- Engenheiro eletricista
- Mestrando em engenharia elétrica
- Líder técnico de Engenharia no FIT
- Possui experiência em Design, montagem e fabricação de PCB, desenvolvimento web, nuvem e software para embarcados
- CTO da LabMaker.Tech



/eng-viniciussouza



# LabMaker.Tech

Desenvolve:

**Hardware** (Design de circuito eletrônico, Design de PCB, Fabricação de PCB e Montagem de PCBA)

**Firmware** (Desenvolvimento de software para embarcados)

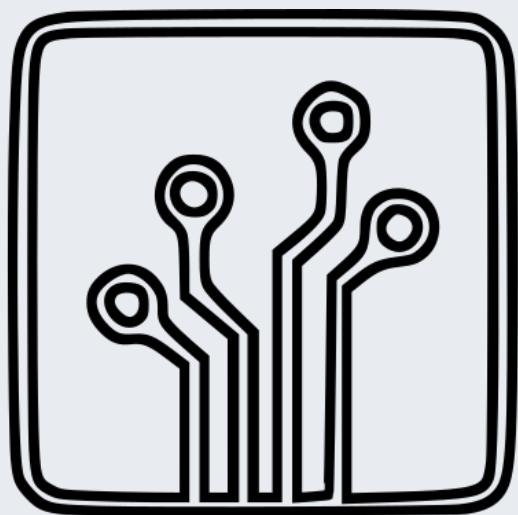
**Software** (Desenvolvimento web, nuvem e aplicativo)

**Impressão 3D** (FDM, SLA e SLS)

**Robótica industrial e educacional**

**Consultoria** (Hardware, Firmware, Software, Impressão 3D e Robótica em geral)

**Materiais didáticos** (Kits de eletrônica, apostilas e vídeos)



LabMaker.Tech



/LabMaker.Tech

# Índice

1. O que é IoT (Internet das Coisas)
2. Impressão 3D
3. Exemplos de Impressão 3D em projetos IoT
4. Conhecendo o ESP32-S3 Zero
5. Desenvolvimento de um Projeto IoT
6. Modelagem de um Case para o Projeto
7. Montagem do Projeto Completo
8. Exemplos de Aplicação do Projeto
9. Encerramento

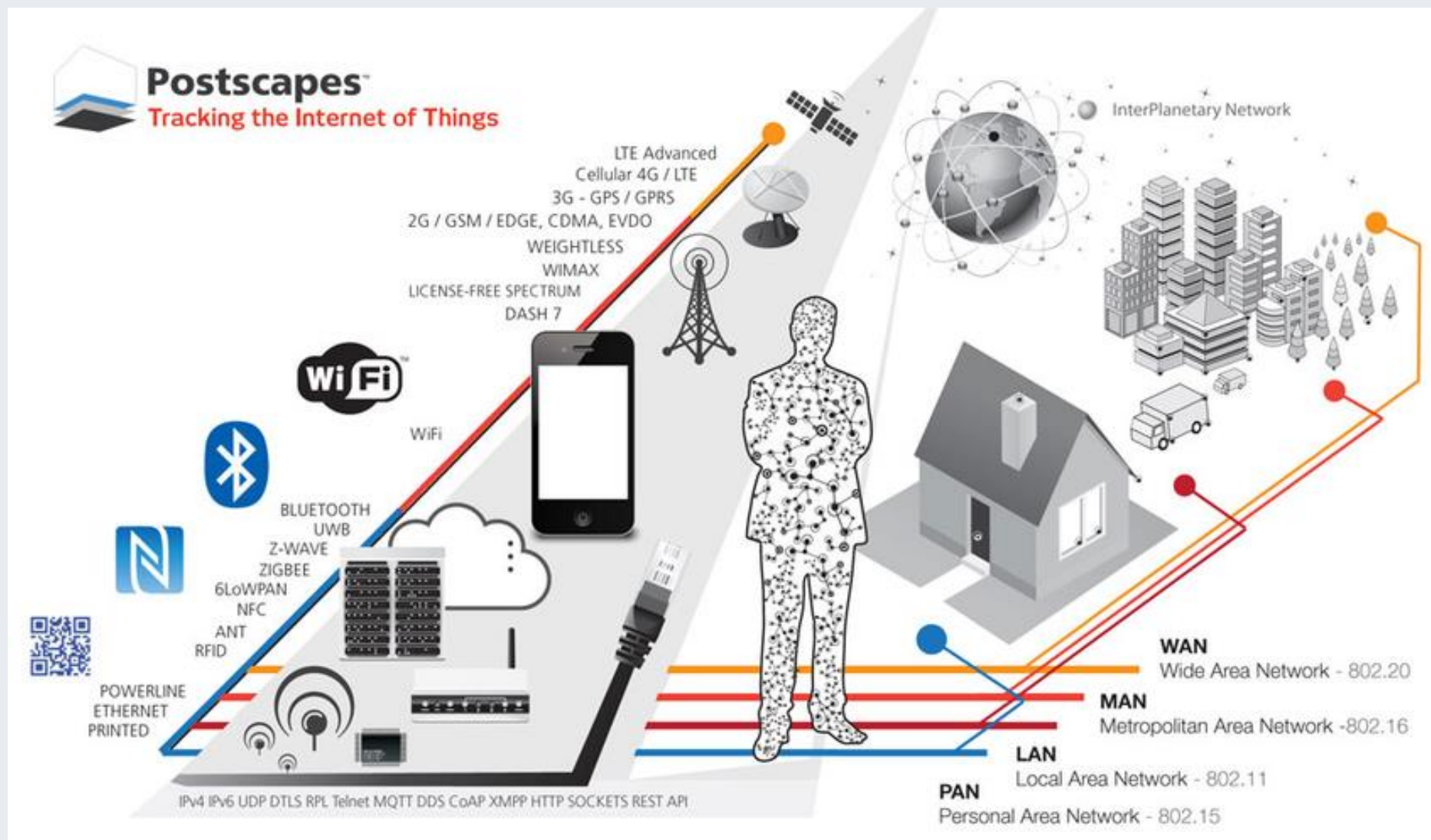
# O que é IoT (Internet das Coisas)





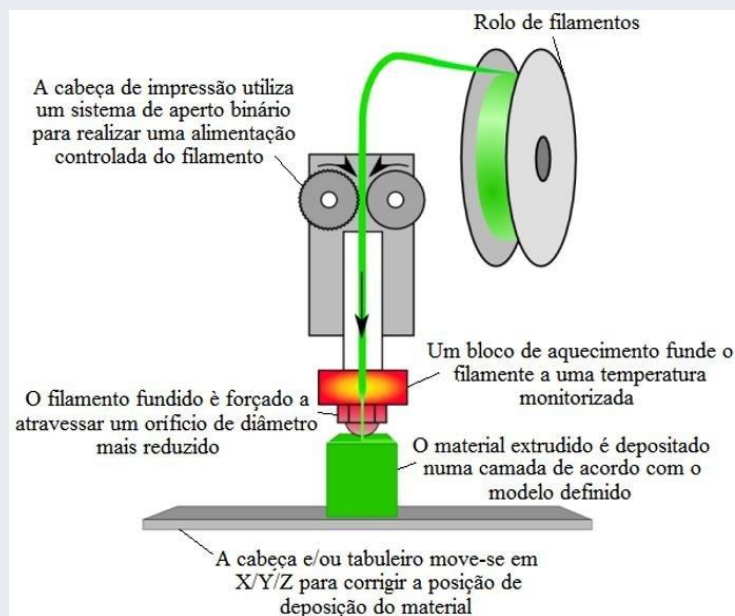


# O que é IoT (Internet das Coisas)

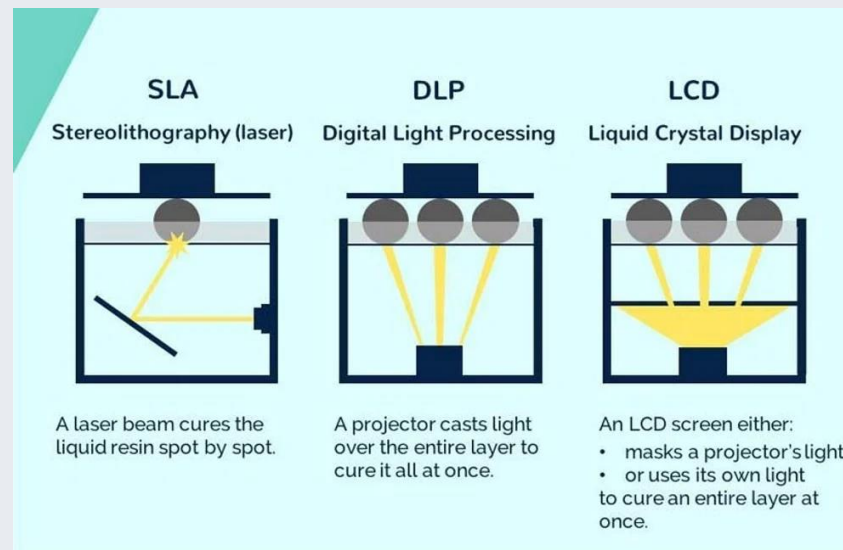




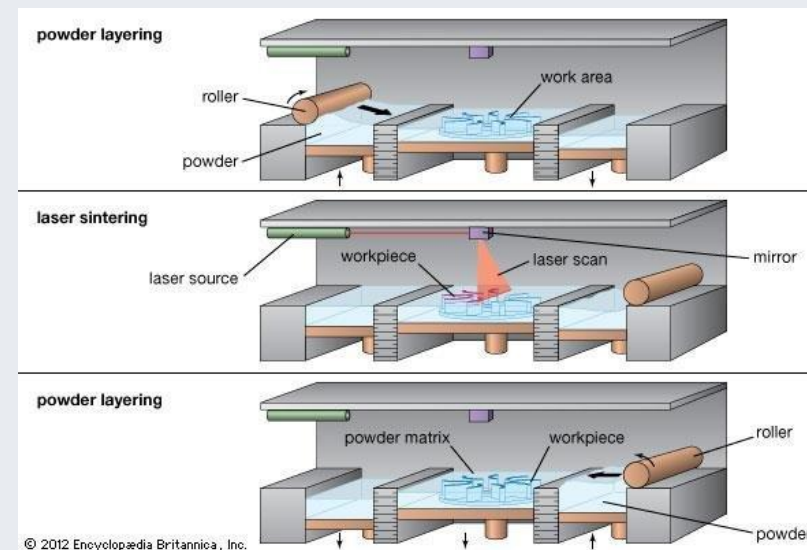
# Impressão 3D



FDM



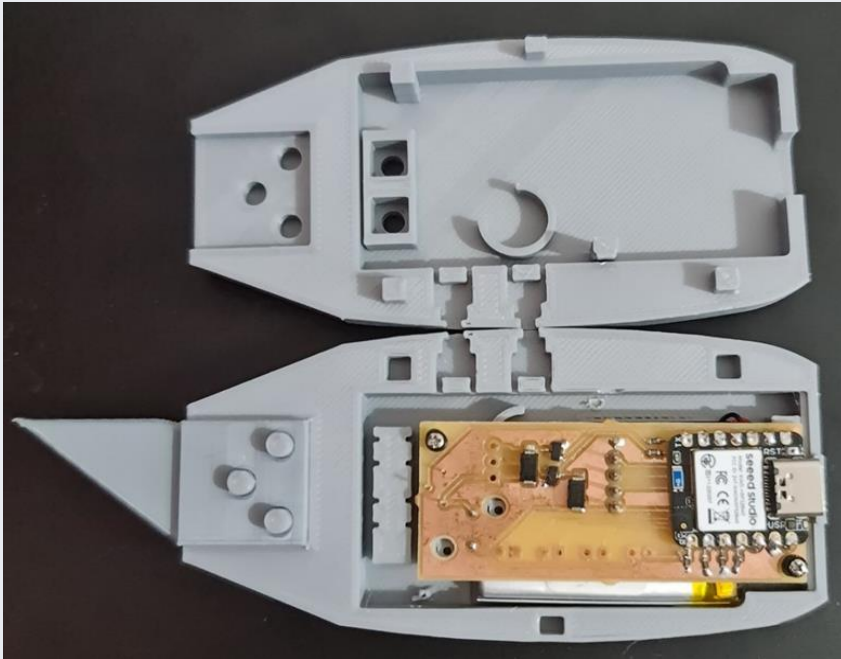
SLA/DLP/LCD



SLS



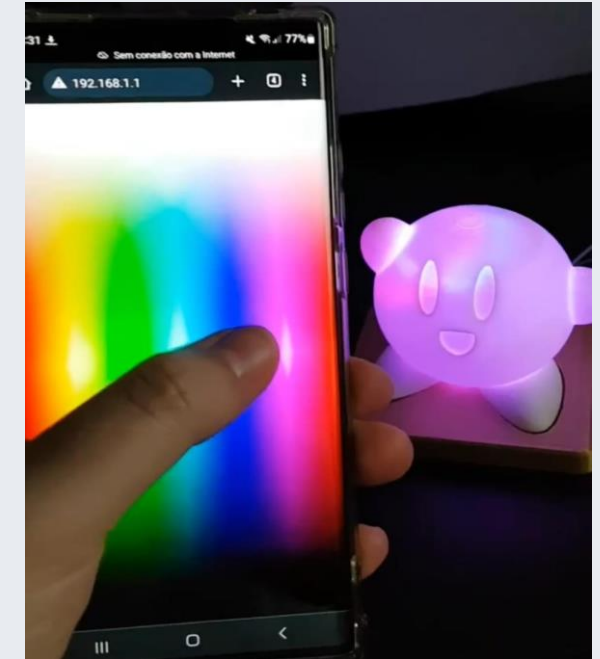
# Exemplos de Impressão 3D em projetos IoT



Bisturi para ambiente de realidade virtual  
Comunicação: BLE  
Impressão 3D: FDM ou MSLA  
Desenvolvedor: LabMaker.Tech



Faixa para captura de EEG  
Comunicação: BLE  
Impressão 3D: SLS  
Desenvolvedor: SleepUp



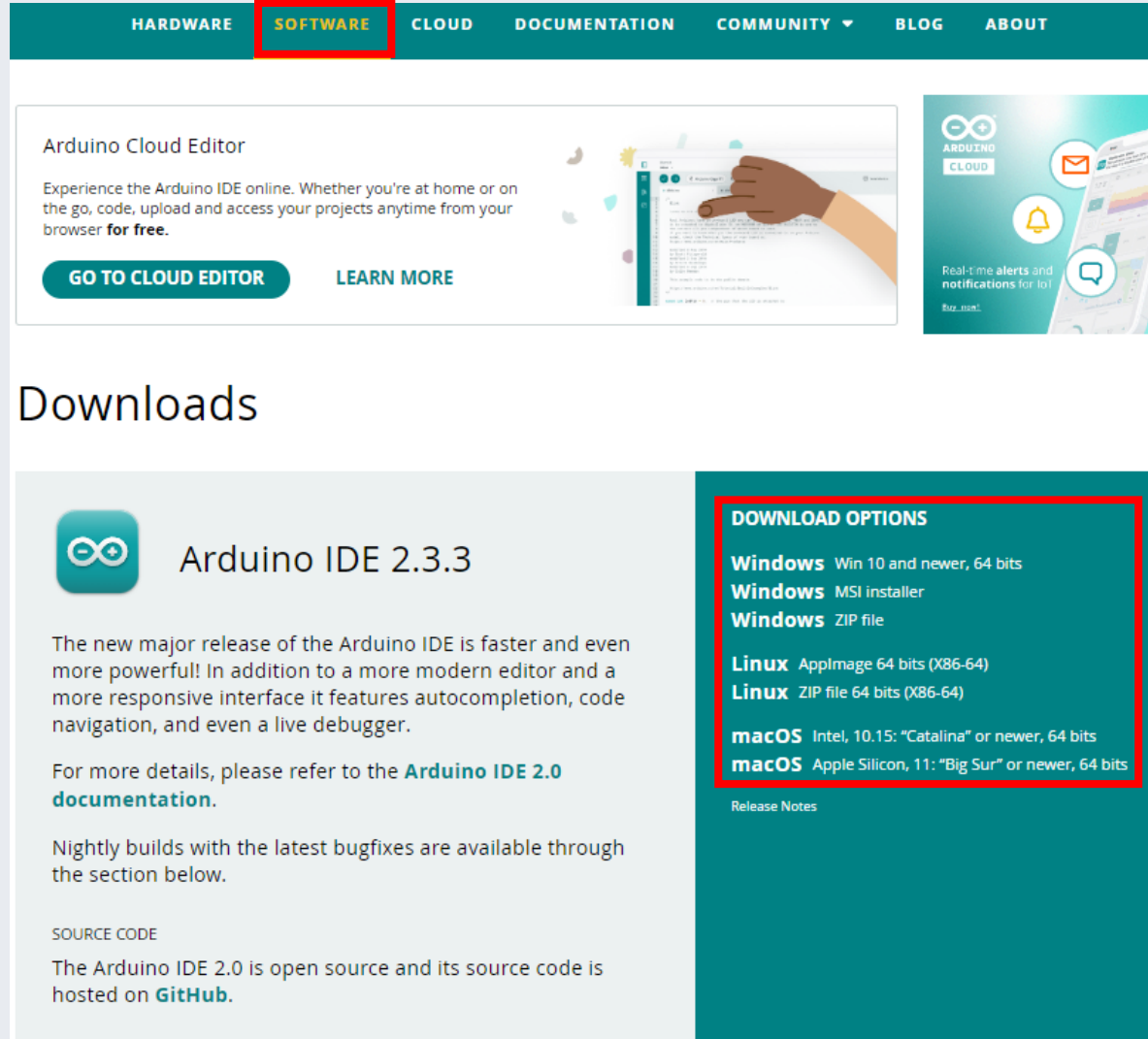
Boneco Kirby com ajuste de cores  
Comunicação: WiFi  
Impressão 3D: FDM  
Desenvolvedor: LabMaker.Tech



# Conhecendo o ESP32-S3 Zero



# Baixando e instalando o Arduino IDE



The screenshot shows the Arduino.cc website with the 'SOFTWARE' tab highlighted in the navigation bar. Below the navigation bar, there's a section for 'Arduino Cloud Editor' with a 'GO TO CLOUD EDITOR' button. The main heading is 'Downloads'. Under 'Downloads', there's a section for 'Arduino IDE 2.3.3' with a description of the new major release. To the right of this section, there's a 'DOWNLOAD OPTIONS' box with a red border, listing download options for Windows, Linux, and macOS. The options are: Windows (Win 10 and newer, 64 bits), Windows (MSI installer), Windows (ZIP file), Linux (AppImage 64 bits (X86-64)), Linux (ZIP file 64 bits (X86-64)), macOS (Intel, 10.15: "Catalina" or newer, 64 bits), and macOS (Apple Silicon, 11: "Big Sur" or newer, 64 bits). Below the download options, there's a link for 'Release Notes'.


**SOFTWARE** CLOUD DOCUMENTATION COMMUNITY ▼ BLOG ABOUT

**Arduino Cloud Editor**

Experience the Arduino IDE online. Whether you're at home or on the go, code, upload and access your projects anytime from your browser **for free**.

[GO TO CLOUD EDITOR](#) [LEARN MORE](#)

## Downloads

 **Arduino IDE 2.3.3**

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through the section below.

**SOURCE CODE**

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

**DOWNLOAD OPTIONS**

**Windows** Win 10 and newer, 64 bits  
**Windows** MSI installer  
**Windows** ZIP file

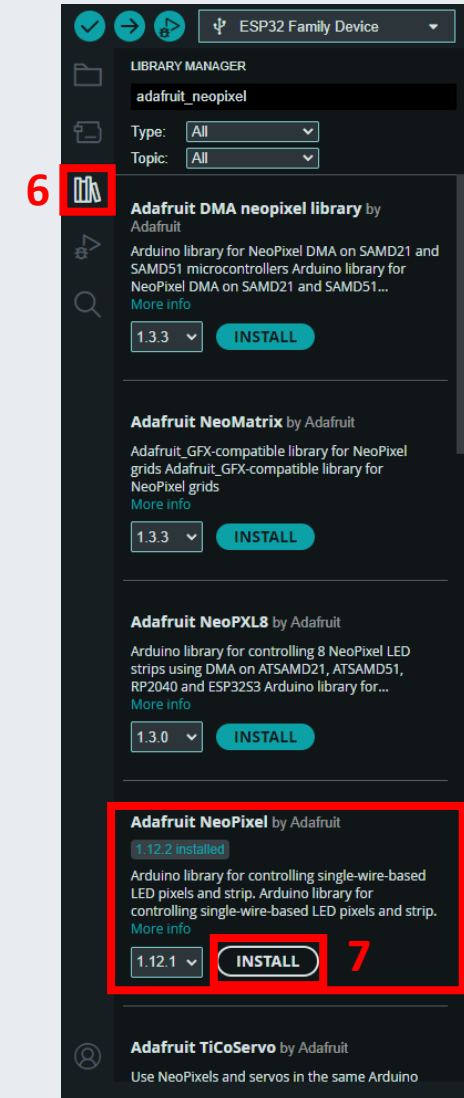
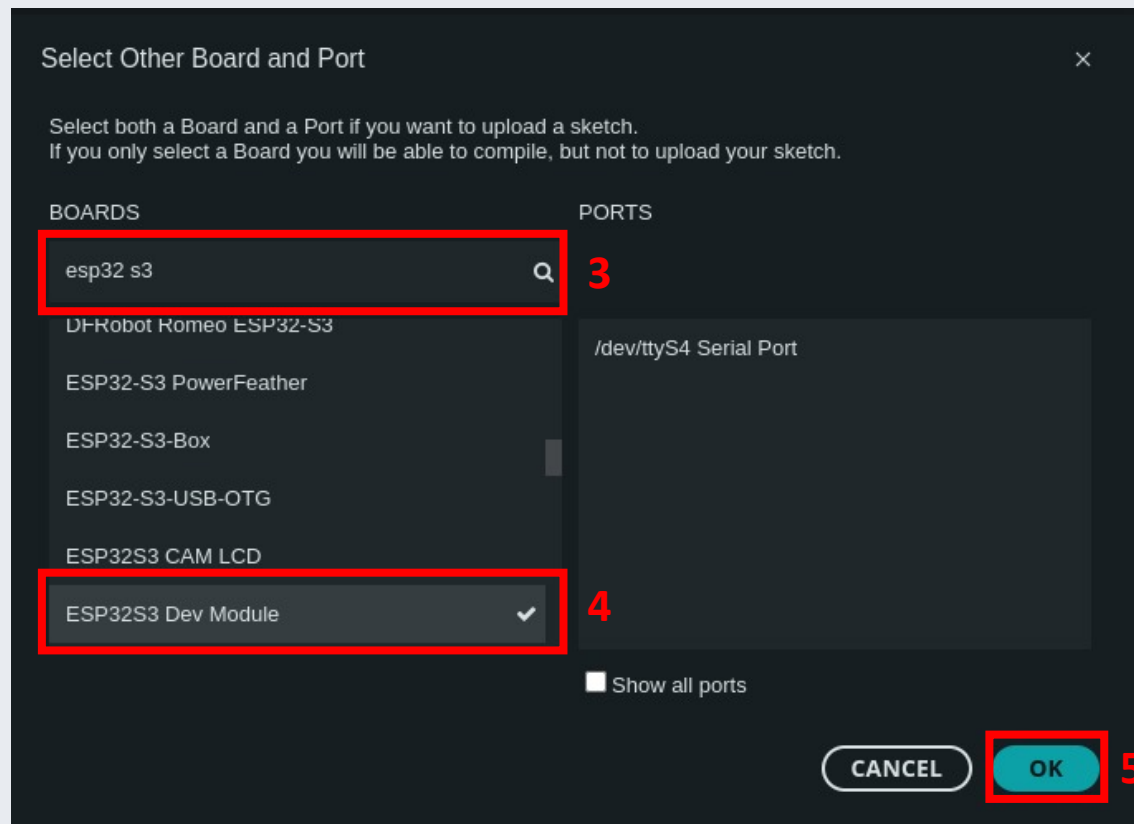
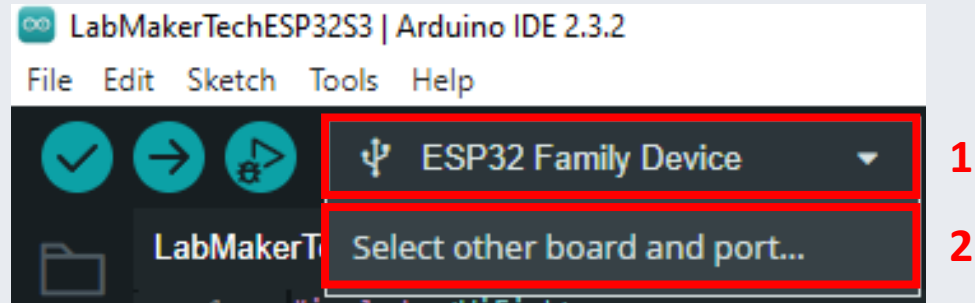
**Linux** AppImage 64 bits (X86-64)  
**Linux** ZIP file 64 bits (X86-64)

**macOS** Intel, 10.15: "Catalina" or newer, 64 bits  
**macOS** Apple Silicon, 11: "Big Sur" or newer, 64 bits

[Release Notes](#)

<https://www.arduino.cc/en/software>

# Instalando recursos Arduino







# Desenvolvimento de um Projeto IoT

```
1 #include <WiFi.h>
2 #include <WebServer.h>
3 #include <Adafruit_NeoPixel.h>
4
5 // Definições para o WS2812
6 #define LED_PIN 21 // Pino onde o WS2812 está conectado
7 #define NUMPIXELS 1 // Número de LEDs WS2812
8
9 Adafruit_NeoPixel pixels(NUMPIXELS, LED_PIN, NEO_GRB + NEO_KHZ800);
10
11 // Credenciais da rede Wi-Fi
12 const char* ssid = "ALTERAR-REDE";
13 const char* password = "ALTERAR-SENHA";
14
15 // Cria o servidor web na porta 80
16 WebServer server(80);
17
18 // Função para configurar o LED WS2812
19 void setupPixels() {
20     pixels.begin(); // Inicializa o NeoPixel
21     pixels.show(); // Apaga o LED
22 }
23
24 // Função para atualizar a cor do LED
25 void updateLED(int r, int g, int b) {
26     pixels.setPixelColor(0, pixels.Color(r, g, b)); // Define a cor RGB
27     pixels.show(); // Atualiza o LED para mostrar a nova cor
28 }
```

```
30 // Função que será chamada quando o servidor web for acessado
31 void handleRoot() {
32     String html = "<html><head><title>Controle RGB WS2812</title>";
33     html += "<style>";
34     html += "body { font-family: Arial; text-align: center; background-color: #1E3A5F; color: white; }; // Define o fundo azul
35     html += "h1 { color: #FFD700; margin-bottom: 5px; }; // Título principal dourado
36     html += "h2 { color: black; margin-top: 0px; }; // Nome da aplicação em preto
37     html += "input[type=range] { width: 300px; height: 30px; }; // Aumenta o tamanho dos sliders
38     html += ".slider-red { background-color: red; };
39     html += ".slider-green { background-color: green; };
40     html += ".slider-blue { background-color: blue; };
41     html += "#colorDisplay { width: 100px; height: 100px; border-radius: 50%; margin: 20px auto; background-color: rgb(0,0,0); };
42     html += "</style>";
43     html += "<script>";
44     html += "function updateColor() {";
45     html += "    var r = document.getElementById('r').value;";
46     html += "    var g = document.getElementById('g').value;";
47     html += "    var b = document.getElementById('b').value;";
48     html += "    document.getElementById('colorDisplay').style.backgroundColor = 'rgb(' + r + ',' + g + ',' + b + ')';";
49     html += "    document.getElementById('r').style.background = 'rgb(' + r + ',0,0)'; // Atualiza cor do slider vermelho
50     html += "    document.getElementById('g').style.background = 'rgb(0,' + g + ',0)'; // Atualiza cor do slider verde
51     html += "    document.getElementById('b').style.background = 'rgb(0,0,' + b + ')'; // Atualiza cor do slider azul
52     html += "};";
53     html += "</script></head>";
54     html += "<body onload='updateColor()'>";
55
56     // Título principal
57     html += "<h1>LabMaker.Tech</h1>"; // Título dourado
58     html += "<h2>Controle RGB WS2812</h2>"; // Subtítulo em preto
59
60     // Sliders
61     html += "<form action='\"/setRGB\"' oninput='updateColor()'>";
62     html += "R: <input type='\"range\"' class='\"slider-red\"' id='\"r\"' name='\"r\"' min='\"0\"' max='\"255\"' value='\"0\"'><br>";
63     html += "G: <input type='\"range\"' class='\"slider-green\"' id='\"g\"' name='\"g\"' min='\"0\"' max='\"255\"' value='\"0\"'><br>";
64     html += "B: <input type='\"range\"' class='\"slider-blue\"' id='\"b\"' name='\"b\"' min='\"0\"' max='\"255\"' value='\"0\"'><br>";
65
66     // Círculo para mostrar a cor
67     html += "<div id='colorDisplay'></div>";
68
69     html += "<input type='\"submit\"' value='\"Atualizar\"'>";
70     html += "</form></body></html>";
71
72     server.send(200, "text/html", html);
73 }
```



# Desenvolvimento de um Projeto IoT

```
75 // Função para ler os valores RGB do formulário e atualizar o LED
76 void handleSetRGB() {
77     int r = server.arg("r").toInt();
78     int g = server.arg("g").toInt();
79     int b = server.arg("b").toInt();
80
81     updateLED(r, g, b);
82
83     server.sendHeader("Location", "/");
84     server.send(303);
85 }
86
87 void setup() {
88     Serial.begin(115200);
89     setupPixels();
90
91     // Conecta ao Wi-Fi
92     WiFi.begin(ssid, password);
93     while (WiFi.status() != WL_CONNECTED) {
94         delay(1000);
95         Serial.println("Conectando ao WiFi...");
96     }
97     Serial.println("Conectado ao WiFi");
98
99     // Configura as rotas do servidor web
100    server.on("/", handleRoot);
101    server.on("/setRGB", handleSetRGB);
102
103    // Inicia o servidor
104    server.begin();
105    Serial.println("Servidor web iniciado");
106 }
107
108 void loop() {
109     server.handleClient(); // Processa as requisições do cliente
110 }
```



/vinihsouza/LabMakerTechESP32S3

# Modelagem de um Case para o Projeto



# Exemplos de Aplicação do Projeto

**Automação residencial (ex.: controle de iluminação):** Utilizando o ESP32-S3 e o controle via webserver, é possível controlar a iluminação da casa remotamente, permitindo o ajuste da cor do LED de acordo com o ambiente ou criar cenas automatizadas.



**Projetos educacionais e ensino de eletrônica:** Este projeto pode ser utilizado em salas de aula para ensinar conceitos de IoT, programação e eletrônica para estudantes. A possibilidade de controlar o LED RGB torna a experiência interativa e visualmente impactante.

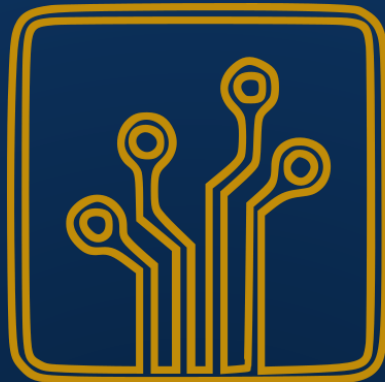


**Uso em *wearables* e dispositivos de monitoramento remoto:** Pequenos dispositivos vestíveis podem ser construídos com LEDs RGB e controlados via Bluetooth ou Wi-Fi para diferentes finalidades, como *fitness trackers* ou acessórios de moda tecnológica.



**Dispositivos de monitoramento remoto:** O ESP32-S3 pode ser usado em aplicações de monitoramento ambiental (ex.: temperatura e umidade), esses dados podem ser visualizados via um painel web (dashboard), utilizando o LED RGB como indicador de condições (ex.: verde para "normal" e vermelho para "alerta").





LabMaker.Tech

*Obrigado!*

